# AI Infrastructure Security Playbook

## Google Cloud (GCP) Implementation Guide

Cloud-Specific Controls Mapped to ETSI EN 304 223 Risk Gradient L1–L8

V1.0 — February 2026

Risk-tiered · Capability-based · Exposure-driven


https://deepcyber.ai

# Contents




© Deep Cyber Ltd 2026

# 1. Introduction

This guide provides Google Cloud (GCP)-specific implementation instructions for the AI Infrastructure Security Playbook. It maps each control from the playbook's seven dimensions (Governance, Identity, Data, Processing, Network, Supply Chain, Monitoring and Operations) to concrete Google Cloud (GCP) services, configurations, and recommended settings. All controls are cumulative: higher tiers inherit and extend lower-tier controls. Controls must be validated via automated policy enforcement where technically feasible.

This guide should be read alongside the main AI Infrastructure Security Playbook, which defines the Risk Gradient (L1–L8), cross-cutting risk modifiers, and the ETSI EN 304 223 alignment rationale. This document focuses on the how rather than the what and why.

This guide assumes a GCP environment with Organization Policy Service, Resource Manager folder hierarchy, and Google Workspace or Cloud Identity for identity management. Controls reference GCP-native services by default.

For Security Command Center, enable the Premium tier for comprehensive security findings, compliance monitoring, and threat detection. For Vertex AI workloads, apply Organization Policy constraints for machine type and region restrictions.


© Deep Cyber Ltd 2026

# 2. Getting Started

Before working through the detailed control checklists, follow these five steps to establish your baseline and create a prioritised action plan. This sequence moves from discovery through assessment to actionable remediation, ensuring you focus effort where risk is highest.

## Step 1: Build Your AI Inventory

You cannot secure what you do not know exists. Begin by discovering and documenting every AI workload, tool, and service in use across your organisation. Shadow AI is often the largest unmanaged risk.

- ☐ Conduct a discovery exercise across all business units to identify every AI tool, service, and automation in use — including embedded features (Copilot, Gemini, Atlassian Intelligence), citizen developer automations, API integrations, and ML workloads
- ☐ Classify each AI workload against the Risk Gradient (L1–L8) using the tier definitions in the main playbook
- ☐ For each workload, assess the five cross-cutting risk modifiers (Exposure, Data Sensitivity, Autonomy, Integration Privilege, Physical-World Effect) to determine the effective risk level
- ☐ Document the data sensitivity classification for each workload: what data does it access, process, store, or generate? Include both designed access and potential access given current permissions
- ☐ Assign an owner to each AI workload who is accountable for its security posture
- ☐ Record all entries in a centralised AI Inventory (spreadsheet, CMDB, or dedicated AI governance tool) that is maintained as a living document

## Step 2: Assess Your Current Security Baseline with CSPM

For existing AI workloads, your Cloud Security Posture Management (CSPM) tooling provides an immediate, evidence-based view of your current gaps. Enable Google Security Command Center (SCC) Premium tier. Enable SCC across the entire organisation. Use Security Health Analytics for automated misconfiguration detection, Event Threat Detection for runtime threats, and Container Threat Detection for GKE workloads. Apply SCC custom modules to create findings specific to AI workload misconfigurations and review findings across all environments hosting AI workloads.

- ☐ Review findings and ensure the following baseline controls are in place for every AI workload:
  - (a) Encryption at rest and in transit is enforced on all AI data stores, model artifacts, and communication channels
  - (b) Identity and access policies follow least privilege — no overly permissive roles on AI resources
  - (c) Network exposure is minimised — no unintended public endpoints on AI services
  - (d) Logging and monitoring are enabled for all AI workloads
  - (e) Vulnerability scanning is active on all compute, containers, and dependencies
- ☐ Record your CSPM secure score or compliance percentage as your starting baseline for AI workloads
- ☐ Export the CSPM findings for your AI workloads for use in Step 3

GCP-specific CSPM setup:

- ☐ Enable SCC Premium across the entire GCP Organisation via the Security Command Center API
- ☐ Enable Security Health Analytics and Event Threat Detection for all projects hosting AI workloads
- ☐ Enable Container Threat Detection for GKE clusters running AI agents or inference workloads
- ☐ Use Cloud Asset Inventory to discover and tag all AI-related resources across projects
- ☐ Review SCC Security Health Analytics findings: record the baseline vulnerability and misconfiguration count for AI projects
- ☐ Export SCC findings via Pub/Sub or BigQuery for ingestion by the DeepCyber ETSI Agent in Step 3

## Step 3: Map Findings to ETSI EN 304 223 with the DeepCyber ETSI Agent

Raw CSPM findings tell you what is misconfigured, but not how it maps to AI-specific security requirements. The DeepCyber ETSI Agent bridges this gap by ingesting your CSPM findings, mapping them against ETSI EN 304 223 provisions, and generating a tailored, prioritised remediation checklist aligned to your assessed risk gradient level.


© Deep Cyber Ltd 2026

- ☐ Feed your exported CSPM findings into the DeepCyber ETSI Agent along with your AI Inventory and risk gradient assessments from Step 1
- ☐ The agent maps each finding to the relevant ETSI EN 304 223 principle and provision, identifies which risk gradient levels are affected, and prioritises remediation by risk severity and blast radius
- ☐ Review the generated checklist: it will highlight quick wins (controls that close multiple ETSI provisions simultaneously) and critical gaps (controls missing at your highest risk gradient levels)
- ☐ Use the prioritised output to create your remediation backlog, assigning owners and target dates to each action item

## Step 4: Remediate by Priority

Work through the prioritised remediation checklist, starting with critical findings at your highest effective risk gradient levels. Focus on controls that deliver the greatest risk reduction per unit of effort.

- ☐ Address critical and high-severity findings first, prioritising workloads at the highest effective risk gradient level
- ☐ Target quick wins that close multiple ETSI provisions simultaneously: enforcing encryption (P6), enabling MFA (P6), activating logging (P12), and applying least-privilege access (P5, P6)
- ☐ Use the detailed control checklists in this guide as the reference for implementing each control
- ☐ Track remediation progress against your CSPM secure score and ETSI provision coverage percentage

## Step 5: Establish Continuous Governance

AI security is not a one-time exercise. New AI workloads are deployed continuously, cloud configurations drift, and new threats emerge. Establish recurring processes to maintain your security posture over time.

- ☐ Schedule recurring CSPM reviews (minimum monthly) across all AI workload environments
- ☐ Re-run the DeepCyber ETSI Agent periodically (quarterly recommended) to reassess posture against ETSI provisions as your environment evolves
- ☐ Update the AI Inventory whenever new AI workloads are deployed, existing workloads change scope, or workloads are decommissioned
- ☐ Integrate AI workload onboarding into existing change management processes: require risk gradient assessment and security review before any new AI workload enters production
- ☐ Report AI security posture to leadership quarterly using CSPM scores, ETSI provision coverage, and remediation velocity as key metrics


© Deep Cyber Ltd 2026

© Deep Cyber Ltd 2026

## 2A. How to Use This Playbook

This playbook is designed for multiple audiences. Use the guide below to find the fastest path to the content most relevant to your role.

| Your Role | What to Do | Start Here |
|---|---|---|
| **CISO / Security Leader** | Review the service mapping table and minimum baseline to understand your cloud posture requirements. | Section 3 (Service Mapping), then Minimum Baseline table in Section 3A |
| **Platform / Cloud Engineer** | Use the tier checklists as implementation runlists. Each control names the specific cloud service and configuration. | Section 2 (Getting Started), then Section 4 (Implementation Checklists) |
| **Security Architect** | Map the service table to your existing cloud architecture and identify gaps per tier. | Section 3 (Service Mapping), then Section 4 (Checklists) |
| **GRC / Compliance** | Use the checklists as evidence of control implementation against ETSI EN 304 223 provisions. | Section 2 (Getting Started, Step 3: DeepCyber ETSI Agent), then Section 4 (Checklists) |

*Tip: All readers should begin with the Getting Started workflow (five steps) to build an AI inventory and establish a baseline before diving into detailed controls.*

# 3. Google Cloud (GCP) Service Mapping

The following table maps each control dimension to the primary Google Cloud (GCP) services used to implement the playbook controls.

| Control Dimension | Primary Services | Key Configuration |
|---|---|---|
| **Governance** | Organization Policy Service, Security Command Center (SCC), Assured Workloads, Resource Manager | Deploy Org Policy constraints restricting AI service enablement; enable SCC Premium for AI security findings; use labels for AI resource classification |
| **Identity** | Cloud IAM, Workload Identity Federation, Secret Manager, Cloud KMS, IAM Conditions | Enforce Workload Identity for all AI workloads; store secrets in Secret Manager with rotation; use Cloud KMS CMEKs; apply IAM Conditions |
| **Data** | Cloud DLP, Data Catalog, BigQuery column-level security, Cloud Storage encryption | Deploy DLP API inspection on AI data flows; enforce CMEK on all AI data stores; use Data Catalog for lineage; apply VPC Service Controls |
| **Processing** | Cloud Run, GKE, Apigee, Cloud Build, Vertex AI, Cloud Functions | Deploy AI workloads on Cloud Run/GKE; route AI APIs through Apigee; enforce Cloud Build pipeline gates |
| **Network** | VPC, Firewall Rules, Private Google Access, Cloud Armor, VPC Service Controls, Private Service Connect | Isolate AI workloads in dedicated VPCs; use Private Service Connect for Vertex AI; deploy Cloud Armor WAF; enable VPC Service Controls |
| **Supply Chain** | Artifact Registry, Binary Authorization, Container Analysis, Software Delivery Shield | Enable Artifact Registry scanning; enforce Binary Authorization for GKE deployments; use Software Delivery Shield for SBOM |
| **Monitoring & Ops** | Security Command Center, Chronicle, Cloud Logging, Cloud Monitoring, Cloud Audit Logs | Enable SCC Premium; deploy Chronicle for SIEM; stream Cloud Audit Logs centrally; set Cloud Monitoring alerting policies |


© Deep Cyber Ltd 2026

## 3A. Minimum Baseline Controls by Tier

The following table summarises the minimum required infrastructure controls at each risk gradient level. Use this as a quick-reference before working through the detailed checklists. Controls are cumulative: each tier inherits all controls from lower tiers.

| Tier | Pattern | Minimum Required Controls (cumulative) |
|---|---|---|
| L1 | **Embedded AI** | MFA on all AI-enabled accounts, DLP integration with AI endpoints, AI acceptable use policy, CSPM enabled, prompt/response logging, shadow AI discovery, quarterly access reviews |
| L2 | **AI-Assisted Dev** | + Mandatory code review for AI-generated code, SAST/DAST in CI/CD, AI code provenance tracking, developer training on AI code risks, dependency scanning on AI-suggested packages |
| L3 | **Citizen Agents** | + Environment separation (dev/prod), connector permission reviews, DLP on low-code connectors, workflow approval gates, JIT access for agent service accounts, agent inventory register |
| L4 | **API Orchestration** | + API gateway with rate limiting, credential vault (no hardcoded keys), network segmentation for AI services, input validation on all API endpoints, egress filtering, API key rotation policy |
| L5 | **Autonomous Agents** | + Tool call allowlisting, agent sandboxing, memory TTL enforcement, human-in-the-loop for high-risk actions, permission boundaries preventing self-escalation, kill switches |
| L6 | **ML Pipelines** | + Data provenance and lineage tracking, artifact signing and integrity verification, training environment isolation, model registry access controls, pipeline audit logging, SBOM for model dependencies |
| L7 | **Model Hosting** | + Inference endpoint rate limiting, model extraction detection, adversarial input filtering, private endpoints (no public exposure), model versioning with rollback, A/B deployment gates |
| L8 | **Multi-Agent** | + mTLS between agents, capability-scoped identity per agent, trust boundary enforcement, circuit breakers, cascade failure detection, real-time agent behaviour monitoring, autonomous privilege escalation prevention |

Note: "+" indicates controls added at this tier, in addition to all controls inherited from lower tiers. The detailed checklists in the following section provide the full implementation specification.

### Control Dimensions at a Glance

Each dimension applies at every tier. The table shows what changes as you move up the risk gradient.

| Dimension | At L1 (Foundational) | At L8 (High Assurance) | ETSI |
|---|---|---|---|
| **Governance** | AI acceptable use policy, shadow AI audits | AI safety board, autonomous system risk review, kill-switch governance | P1, P3, P4 |
| **Identity** | MFA, Conditional Access on licences | mTLS, capability-scoped agent identity, JIT with session tokens | P4, P5, P6 |
| **Data** | DLP on prompts, data classification | Provenance tracking, lineage audit, cross-agent data flow controls | P5, P8, P13 |
| **Processing** | Tenant config review, feature toggles | Agent sandboxing, capability tokens, tool-call enforcement | P2, P6, P9 |
| **Network** | Existing segmentation sufficient | Service mesh, private endpoints, per-agent egress rules, circuit breakers | P2, P6 |
| **Supply Chain** | Vendor DPA review | Model SBOM, artifact signing, training data provenance, dependency pinning | P7 |
| **Monitoring** | CSPM, basic alerting | Real-time agent behaviour monitoring, cascade detection, anomaly ML | P11, P12 |


© Deep Cyber Ltd 2026

Same seven dimensions at every tier – radically different infrastructure controls. The detailed checklists specify every control.

## 3C. Common AI Infrastructure Failure Patterns

AI infrastructure failures are rarely model failures – they are identity, network, and supply chain failures amplified by AI capability. The following patterns represent the most common infrastructure-level failures observed in AI deployments. Each maps directly to controls in the tier checklists that follow.

| Failure Pattern | What Happens | Controls That Prevent It |
|---|---|---|
| **Over-permissioned agent with production write access** | A citizen-built Power Automate agent granted broad connector permissions writes directly to production HR or finance systems. No approval gate, no audit trail. | L3+: Governance, Identity |
| **LLM API key committed to a public repository** | An API key for a paid LLM service is hardcoded in source and pushed to GitHub. Automated scanners find it within minutes. The key has no spending cap or IP restriction. | L4+: Identity, Supply Chain |
| **Fine-tuned model promoted without integrity verification** | A model trained on sensitive data is promoted from staging to production with no cryptographic hash check. A tampered artifact enters the serving pipeline undetected. | L6+: Processing, Supply Chain |
| **Inference endpoint exposed without rate limiting** | A self-hosted LLM endpoint is deployed on a public subnet with no WAF, no rate limit, and no query pattern monitoring. Systematic extraction queries exfiltrate the model weights. | L7+: Network, Processing, Monitoring |
| **Static service account shared across agent mesh** | Multiple agents in a multi-agent system share a single service account with broad IAM permissions. One compromised agent inherits the access of all others. | L8: Identity, Governance |
| **Cross-agent implicit trust enabling lateral compromise** | Agents in a multi-agent system accept task delegations from any peer without verifying identity or capability. An injected prompt in one agent propagates through the mesh. | L8: Processing, Network, Identity |
| **Shadow AI tool adopted without security review** | A team enables an AI meeting transcription service and grants it access to calendar and email. No DPIA, no DLP, no vendor security review. Sensitive data flows to an unvetted third party. | L1+: Governance, Data |
| **AI-generated code deployed without review** | A developer uses AI code completion to generate database queries. The AI hallucinates a dependency and introduces an SQL injection vulnerability. No human review, no SAST scan. | L2+: Processing, Supply Chain |

Every pattern above was preventable with controls already in this playbook. The tier checklists that follow provide the specific implementation steps.


© Deep Cyber Ltd 2026

# 4. Implementation Checklists by Risk Gradient Level

Each tier's controls are mapped to specific Google Cloud (GCP) services with actionable configuration items. Items marked with □ are implementation checkpoints.

## 4.1 L1: Embedded AI

**Example:** Gemini for Google Workspace, Duet AI in Cloud Console, Google Meet AI features

### Governance

- □ Use Google Workspace Admin Console to control Gemini feature availability by organisational unit (OU)
- □ Deploy Organization Policy constraints restricting which projects can enable AI services (Vertex AI, Gemini API)
- □ Enable SCC Premium with AI-relevant security findings across the organisation
- □ Publish AI Acceptable Use Policy and enforce acceptance via Google Workspace Terms of Service settings
- □ Use Cloud Asset Inventory to discover and inventory all AI service enablement across projects
- □ Apply resource labels (ai-tier:l1, data-classification:internal) to all AI-related resources

### Identity

- □ Enforce Google Workspace 2-Step Verification (2SV) for all users with Gemini access
- □ Control Gemini access via Google Workspace licence assignment by group membership
- □ Apply IAM Conditions restricting AI service admin access by IP range and time
- □ Use Context-Aware Access policies restricting Gemini to managed devices and trusted networks
- □ Review IAM recommendations in SCC to identify overly permissive AI-related bindings

### Data

- □ Deploy Cloud DLP API to inspect data accessible to Gemini features for sensitive data discovery
- □ Verify Gemini for Workspace data processing location and residency settings in Admin Console
- □ Review Google AI data governance documentation: verify no training data retention for enterprise features
- □ Enable Google Workspace Audit Logs for Gemini interaction events
- □ Apply Google Workspace data regions policy where available for Gemini data processing
- □ Configure Gemini context settings to exclude specific Drive folders or Spaces from AI context

### Processing (Apps, APIs, Integration)

- □ Review Google Workspace Gemini settings: disable in applications where not approved
- □ Verify Gemini respects Google Drive sharing permissions and does not surface data from restricted drives
- □ Control Gemini third-party extensions via Google Workspace Marketplace app allowlisting
- □ Disable unapproved AI Marketplace apps at the OU level

### Network

- □ Verify Gemini for Workspace traffic routes via standard Google endpoints (no additional exposure)
- □ Apply Context-Aware Access to restrict Gemini usage to approved network locations
- □ Use BeyondCorp Enterprise policies to control AI feature access based on device and network context
- □ Block access to unapproved third-party AI services via Chrome Enterprise URL filtering

### Supply Chain

- □ Review Google Cloud Trust Centre for Gemini security certifications (SOC 2, ISO 27001)
- □ Monitor Google Workspace release notes for Gemini security and compliance updates
- □ Review Google sub-processor list for Gemini data processing


© Deep Cyber Ltd 2026

- ☐ Verify Google AI responsible use commitments and data handling documentation

## Monitoring and Operations

- ☐ Enable SCC Premium across all projects; review AI-related security findings and recommendations
- ☐ Stream Google Workspace Audit Logs and Cloud Audit Logs to Chronicle for centralised SIEM
- ☐ Create Chronicle detection rules for unusual Gemini interaction patterns
- ☐ Monitor Gemini usage reports in Google Workspace Admin Console
- ☐ Enable Cloud Monitoring alerting policies for AI service API usage spikes
- ☐ Track SCC Security Health Analytics score to measure AI security posture over time


© Deep Cyber Ltd 2026

## 4.2 L2: AI-Assisted Development

**Example:** Gemini Code Assist, Duet AI in Cloud Shell/Cloud Code, AI code gen in IDEs via Google Cloud extensions

### Governance

- ☐ Control Gemini Code Assist access via Google Cloud IAM: grant roles/cloudaicompanion.user to approved groups only
- ☐ Configure Gemini Code Assist settings at org/project level: enable/disable code completion and chat
- ☐ Publish AI code generation policy enforced via Cloud Build trigger requirements
- ☐ Require AI-generated code flagging in commit messages (enforced via Cloud Build pre-submit hook)

### Identity

- ☐ Provision Gemini Code Assist access via Google Groups with IAM role binding
- ☐ Enforce 2SV on all developer accounts with AI tool access
- ☐ Apply IAM Conditions restricting Gemini Code Assist admin settings to platform engineering roles

### Data

- ☐ Configure Gemini Code Assist code citation to identify when suggestions match training data
- ☐ Review Gemini Code Assist data handling: verify code snippets are not retained for model training (enterprise)
- ☐ Deploy Secret Scanner in Cloud Build pipelines to catch hardcoded secrets in AI-generated code
- ☐ Apply Cloud DLP scanning to Cloud Source Repositories or GitHub repos for leaked credentials

### Processing (Apps, APIs, Integration)

- ☐ Integrate Cloud Build with SAST tools (e.g. SonarQube, Semgrep) running on all code including AI-generated
- ☐ Enable Container Analysis for container vulnerability scanning in Cloud Build
- ☐ Enforce Cloud Build approval gates: require manual approval before production deployment triggers
- ☐ Use Artifact Registry as approved package proxy; block direct public registry access from Cloud Build
- ☐ Apply Cloud Source Repository branch protection: require code review with minimum 2 approvals

### Network

- ☐ Route Gemini Code Assist traffic via VPC Service Controls if enterprise proxy required
- ☐ Block AI coding tool endpoints from production Cloud Build workers that do not need them

### Supply Chain

- ☐ Use Artifact Registry to proxy PyPI, npm, and Maven; curate approved packages
- ☐ Enable Container Analysis vulnerability scanning on all Artifact Registry images
- ☐ Pin all dependency versions; use Software Delivery Shield for dependency provenance verification
- ☐ Generate SBOMs using Cloud Build with SLSA attestation via Binary Authorization

### Monitoring and Operations

- ☐ Monitor Container Analysis findings for vulnerability trends in AI-generated code artifacts
- ☐ Stream Cloud Audit Logs for Gemini Code Assist to Chronicle for security event correlation
- ☐ Track Gemini Code Assist usage metrics via Cloud Logging to detect misuse
- ☐ Integrate Artifact Registry vulnerability findings into SCC for centralised tracking


© Deep Cyber Ltd 2026

## 4.3 L3: Citizen Developer Agents

**Example:** AppSheet with AI, Google Workspace Add-ons, Dialogflow CX agents, third-party low-code on GCP

### Governance

- ☐ Use Organization Policy to restrict which projects can deploy Dialogflow, AppSheet, and Cloud Functions
- ☐ Maintain inventory of all citizen-built automations and agents in Cloud Asset Inventory with labels
- ☐ Enforce project separation: citizen developers work in sandbox projects, production requires promotion approval
- ☐ Require human approval steps in Dialogflow CX flows that trigger production writes
- ☐ Apply Cloud Resource Manager label requirements for all citizen developer resources

### Identity

- ☐ Use dedicated service accounts for agent execution with least-privilege IAM roles. Prohibit long-lived service account keys for agents; all agent workloads must use Workload Identity or Workload Identity Federation with short-lived tokens. Alert on any service account key creation events for AI-labelled projects. Apply IAM Deny policies preventing agents from modifying IAM bindings, creating service accounts, generating keys, or altering Workload Identity bindings under any circumstance
- ☐ Store connector credentials in Secret Manager with automatic rotation
- ☐ Apply IAM Conditions: restrict service account usage to specific projects and IPs
- ☐ Enforce service account key management policy: prefer Workload Identity; disable user-managed keys where possible. Enforce organisation policy constraint iam.disableServiceAccountKeyCreation for all AI-labelled projects. Alert on any attempted override via Cloud Audit Logs
- ☐ Use IAM Recommender to identify and remediate overly permissive agent service accounts

### Data

- ☐ Apply Cloud DLP inspection on data flowing through citizen-built automations
- ☐ Enforce CMEK encryption on Firestore, Cloud Storage, and BigQuery accessed by automations
- ☐ Enable Cloud Audit Logs (Data Access) for all resources accessed by citizen automations
- ☐ Use Data Catalog to tag and classify data sources accessible to citizen-built agents

### Processing (Apps, APIs, Integration)

- ☐ Deploy citizen developer Cloud Functions in isolated projects with resource quotas
- ☐ Route automation-to-backend traffic through Apigee with rate limiting and API key validation
- ☐ Apply Cloud Functions max instance limits and timeout restrictions to prevent runaway execution
- ☐ Use Cloud Endpoints or Apigee for schema validation on all backend API calls
- ☐ Test automations against over-privilege scenarios before production project promotion

### Network

- ☐ Deploy citizen developer Cloud Functions in VPC with Serverless VPC Access Connector
- ☐ Apply VPC Firewall Rules restricting egress from citizen automation VPCs
- ☐ Use VPC Service Controls perimeters to prevent data exfiltration from citizen projects
- ☐ Block outbound internet from citizen developer VPCs unless explicitly allowlisted

### Supply Chain

- ☐ Curate approved Dialogflow integrations and AppSheet connectors
- ☐ Scan Cloud Function dependencies with Container Analysis
- ☐ Verify third-party connector publisher identity before enabling in citizen projects

### Monitoring and Operations

- ☐ Enable SCC Premium for citizen developer projects with automated finding notification
- ☐ Stream Cloud Function execution logs and Cloud Audit Logs to Chronicle


© Deep Cyber Ltd 2026

- ☐ Create Cloud Monitoring alerting policies for execution frequency spikes and error rate increases
- ☐ Use IAM Recommender findings to identify dormant or over-permissioned service accounts quarterly


© Deep Cyber Ltd 2026

## 4.4 L4: Data Analytics & API Orchestration

**Example:** Python on Cloud Run calling Vertex AI API, BigQuery ML, Dataflow with LLM enrichment, RAG with Vertex AI Search

### Governance

- ☐ Require threat modelling for pipelines processing confidential data via Vertex AI or third-party AI APIs
- ☐ Set Cloud Billing budgets with alerts on Vertex AI consumption (50%, 80%, 100%)
- ☐ Apply Organization Policy constraints enforcing labelling on all pipeline resources

### Identity

- ☐ Use Workload Identity Federation for all services calling Vertex AI (no service account keys)
- ☐ Store third-party AI API keys in Secret Manager with automatic rotation
- ☐ Apply Vertex AI IAM roles scoped to specific endpoints/models per pipeline service account
- ☐ Never store API keys in environment variables or Cloud Build substitution variables in plaintext
- ☐ Apply IAM Conditions restricting Vertex AI API calls by source IP or VPC Service Controls

### Data

- ☐ Enforce CMEK on Vertex AI, Cloud Storage, BigQuery, and Firestore via Organization Policy
- ☐ Enable Vertex AI safety settings on all Generative AI API calls
- ☐ Enable Cloud Audit Logs (Data Access) for all Vertex AI API interactions
- ☐ Use BigQuery column-level security and row-level security for pipeline data access control
- ☐ Apply Cloud DLP de-identification transforms before sending data to external AI APIs
- ☐ Use Data Catalog for data lineage tracking through AI enrichment pipelines

### Processing (Apps, APIs, Integration)

- ☐ Deploy Apigee as gateway for all AI API calls with rate limiting, quota, and API key management
- ☐ Implement Cloud Run error handling with retry policies and Cloud Tasks dead-letter queues
- ☐ Separate dev, staging, and prod pipeline projects in distinct GCP projects with distinct IAM
- ☐ Version all pipeline code in Cloud Source Repositories or GitHub with branch protection
- ☐ Use Cloud Build with Container Analysis for pipeline CI/CD with security scanning stages

### Network

- ☐ Deploy Vertex AI with Private Service Connect for private API access
- ☐ Apply VPC Service Controls perimeter around all pipeline projects and Vertex AI resources
- ☐ Apply Firewall Rules restricting pipeline Cloud Run/GKE to approved API endpoints only
- ☐ Enable VPC Flow Logs for all pipeline subnets

### Supply Chain

- ☐ Scan all pipeline container images with Artifact Registry vulnerability scanning
- ☐ Pin Vertex AI SDK and all AI client library versions; monitor for security advisories
- ☐ Use Artifact Registry as approved package registry; block direct public access from build
- ☐ Generate SBOMs with SLSA attestations for all pipeline artifacts

### Monitoring and Operations

- ☐ Monitor Vertex AI API usage via Cloud Monitoring: alert on request volume spikes and error rates
- ☐ Track costs via Cloud Billing budgets with Pub/Sub alerts
- ☐ Stream all pipeline logs to Cloud Logging and Chronicle for security investigation
- ☐ Enable SCC Premium for pipeline projects; review AI-specific findings
- ☐ Deploy Cloud Trace for end-to-end latency visibility across pipeline components


© Deep Cyber Ltd 2026

## 4.5 L5: Custom Autonomous Agents

**Example:** LangChain on Cloud Run/GKE, Vertex AI Agent Builder, custom tool-calling agents with Gemini API

### Governance

- ☐ Require formal threat model (STRIDE for AI) before agent production deployment
- ☐ Define graduated autonomy tiers: high-privilege actions require human approval via Cloud Tasks callback
- ☐ Publish Agent Security Standard enforced via Organization Policy on agent hosting projects
- ☐ Conduct agent red-team exercises in isolated dev/test projects

### Identity

- ☐ Assign each agent a unique service account with least-privilege IAM roles scoped to approved tools
- ☐ Enforce tool-call permissions via IAM resource-level policies (not just agent prompt)
- ☐ Implement human approval gates using Cloud Tasks or Workflows human callback steps
- ☐ Store agent credentials in Secret Manager with Workload Identity access
- ☐ Apply IAM deny policies preventing agents from modifying their own IAM bindings
- ☐ Audit all agent service account usage in Cloud Audit Logs

### Data

- ☐ Encrypt agent memory stores (Firestore, Memorystore, Vertex AI Search) with CMEK
- ☐ Apply Firestore Security Rules: agents can only read/write documents in their own collection
- ☐ Implement Firestore TTL policies on memory documents for automatic data expiry
- ☐ Deploy Vertex AI safety filters for agent input/output content filtering
- ☐ Enable Cloud Audit Logs (Data Access) on all agent data stores

### Processing (Apps, APIs, Integration)

- ☐ Deploy agents on Cloud Run with max instance limits or GKE with pod resource quotas
- ☐ Enforce tool-call allowlists validated at deployment time (not runtime prompt)
- ☐ Deploy circuit breakers using Cloud Run timeout limits and Workflows error handlers
- ☐ Version all agent configs and system prompts in version control with mandatory code review
- ☐ Test agents against prompt injection and tool abuse in isolated projects

### Network

- ☐ Deploy agents in VPC with Serverless VPC Access or GKE private cluster
- ☐ Apply VPC Service Controls perimeter around agent project, Vertex AI, and data stores
- ☐ Block all outbound internet from agent environments via Cloud NAT removal and firewall deny rules
- ☐ Implement network-level kill switch: Cloud Function modifying Firewall Rules triggered by SCC finding

### Supply Chain

- ☐ Pin agent framework versions in container images; scan with Artifact Registry and Container Analysis
- ☐ Enforce Binary Authorization requiring signed images for agent deployment to GKE
- ☐ Use Artifact Registry for approved packages; block public registry from Cloud Build

### Monitoring and Operations

- ☐ Log all agent tool calls in structured JSON to Cloud Logging
- ☐ Create Cloud Monitoring alerting policies for agent behavioural anomalies
- ☐ Monitor Firestore/Memorystore memory growth with Cloud Monitoring alerts
- ☐ Enable SCC Premium for all agent projects
- ☐ Deploy canary inputs via Cloud Scheduler + Cloud Functions to verify agent behaviour
- ☐ Stream agent logs to Chronicle for SOC visibility and correlation


© Deep Cyber Ltd 2026

## 4.6 L6: Data Science & ML Pipelines

**Example:** Vertex AI Training, Vertex AI Pipelines, AutoML, Dataflow feature engineering, Vertex AI Feature Store

### Governance

- ☐ Require formal threat model for all Vertex AI training projects covering data poisoning and model supply chain
- ☐ Use Vertex AI Model Registry with approval metadata: require security sign-off before endpoint deployment
- ☐ Deploy Vertex AI Explainability and Fairness tools as mandatory pipeline stages
- ☐ Apply Organization Policy restricting ML compute machine types and regions
- ☐ Maintain model documentation in Vertex AI Model Registry metadata (model cards)

### Identity

- ☐ Apply Vertex AI IAM roles: separate Custom Job Runner, Pipeline Runner, and Model Registry Admin
- ☐ Use Workload Identity for all Vertex AI pipeline and training service accounts
- ☐ Restrict Model Registry write access to Cloud Build pipeline service account only
- ☐ Store external model API keys in Secret Manager with auto-rotation
- ☐ Enforce 2SV on all Vertex AI Workbench users
- ☐ Sign model artifacts: store the approved artifact hash (SHA-256) in Vertex AI Model Registry metadata or release manifest. Before deployment, compute the hash of the serving artifact and compare approved vs deployed. Block promotion on mismatch

### Data

- ☐ Encrypt Vertex AI training data in Cloud Storage and BigQuery with CMEK via Cloud KMS
- ☐ Use Vertex AI Datasets for versioned training data with managed lineage
- ☐ Apply BigQuery column-level security on training datasets
- ☐ Deploy Cloud DLP to scan training data for unintended PII before training
- ☐ Apply Cloud Storage Object Versioning for critical training datasets
- ☐ Enforce Cloud Storage retention policies for training data lifecycle management
- ☐ Enable Cloud Audit Logs (Data Access) on all training data resources
- ☐ Use Vertex AI Feature Store with point-in-time lookups and access auditing

### Processing (Apps, APIs, Integration)

- ☐ Deploy Vertex AI Training in VPC-peered mode with restricted firewall rules
- ☐ Use Vertex AI Pipelines (Kubeflow) for immutable, versioned pipeline definitions in Git
- ☐ Apply Vertex AI custom job resource limits: max runtime, machine type restrictions via Org Policy
- ☐ Scan custom training container images with Artifact Registry scanning and Container Analysis
- ☐ Deploy Vertex AI Model Evaluation as mandatory pipeline stage before Registry promotion
- ☐ Implement model validation gates: accuracy, bias, and fairness tests before human approval

### Network

- ☐ Deploy Vertex AI Training with VPC peering for network isolation
- ☐ Apply VPC Service Controls perimeter around Vertex AI, Cloud Storage, and BigQuery training resources
- ☐ Use Private Service Connect for Vertex AI API access
- ☐ Block internet access from training VMs via firewall deny-all-egress with specific allows
- ☐ Enable VPC Flow Logs on ML workload subnets

### Supply Chain

- ☐ Verify provenance of base models from Vertex AI Model Garden
- ☐ Enable Artifact Registry vulnerability scanning for all training and inference images
- ☐ Pin ML framework versions in Vertex AI custom container definitions


© Deep Cyber Ltd 2026

- ☐ Use Artifact Registry for Python packages; block direct PyPI from training VMs
- ☐ Generate SBOMs with SLSA attestations for all ML containers
- ☐ Store model checkpoints in Cloud Storage with versioning and hash in Model Registry metadata

## Monitoring and Operations

- ☐ Enable Vertex AI Model Monitoring for data drift, prediction drift, and feature attribution
- ☐ Alert on training job resource anomalies via Cloud Monitoring (potential misuse)
- ☐ Stream Vertex AI audit logs and training metrics to Chronicle and Cloud Logging
- ☐ Enable SCC Premium for all ML projects with Vertex AI-specific findings
- ☐ Monitor Model Registry operations in Cloud Audit Logs: alert on promotion and deletion
- ☐ Run periodic model integrity checks (minimum daily): recompute deployed endpoint model artifact hash and compare against Registry-approved hash. Alert and block on mismatch to detect post-deployment drift or tampering. Automate via Cloud Scheduler and Cloud Functions


© Deep Cyber Ltd 2026

## 4.7 L7: Model Hosting & Serving

**Example:** Vertex AI Endpoints (online prediction), Vertex AI batch prediction, self-hosted vLLM/TGI on GKE

### Governance

- ☐ Publish Model Serving Security Standard enforced via Organization Policy on endpoint projects
- ☐ Require security review gate in Cloud Build before Vertex AI endpoint deployment
- ☐ Define rate-limiting policies on Apigee to impede model extraction
- ☐ Maintain endpoint inventory in Vertex AI Model Registry or Cloud Asset Inventory queries
- ☐ Define rollback SLA: use Vertex AI endpoint traffic splitting for instant model rollback

### Identity

- ☐ Apply Vertex AI endpoint IAM: restrict aiplatform.endpoints.predict role to approved service accounts/groups
- ☐ Deploy Apigee with API keys and OAuth 2.0 for per-consumer rate limiting
- ☐ Use Workload Identity for endpoint serving containers accessing GCP resources
- ☐ Audit endpoint invocation events in Cloud Audit Logs

### Data

- ☐ Enforce TLS 1.2+ on all Vertex AI endpoints (default) and GKE Ingress controllers
- ☐ Deploy Vertex AI safety settings for content filtering on prediction endpoints
- ☐ Enable Vertex AI request-response logging for audit and forensics
- ☐ Implement input validation via Apigee request policies: schema, size, content type

### Processing (Apps, APIs, Integration)

- ☐ Verify model artifact integrity: compare Model Registry hash against deployed model
- ☐ Implement canary deployments using Vertex AI endpoint traffic splitting
- ☐ Apply GKE pod security standards (restricted) for self-hosted inference on Kubernetes
- ☐ Implement Apigee policies: rate limiting, spike arrest, quota, IP filtering
- ☐ Test endpoints against adversarial inputs and extraction probes in staging

### Network

- ☐ Deploy Vertex AI endpoints with Private Service Connect for private access
- ☐ Place Apigee in front with Cloud Armor WAF for external consumers
- ☐ Enable Cloud Armor DDoS protection on inference load balancers
- ☐ Apply VPC Service Controls around inference projects
- ☐ Enable VPC Flow Logs on inference subnets

### Supply Chain

- ☐ Verify model artifact provenance from Vertex AI Model Registry before deployment
- ☐ Scan inference images with Artifact Registry scanning and Container Analysis
- ☐ Enforce Binary Authorization requiring signed images for GKE inference deployments
- ☐ Pin serving framework versions and monitor security advisories

### Monitoring and Operations

- ☐ Monitor Vertex AI endpoint metrics (latency, error rates, prediction count) via Cloud Monitoring with auto-scaling
- ☐ Enable Vertex AI Model Monitoring for output distribution drift in production
- ☐ Deploy canary queries via Cloud Scheduler and alert on output deviation
- ☐ Create Cloud Monitoring anomaly detection alerts for extraction attempt patterns
- ☐ Stream Apigee analytics and Vertex AI logs to Chronicle
- ☐ Enable SCC Premium for inference hosting projects


© Deep Cyber Ltd 2026

© Deep Cyber Ltd 2026

## 4.8 L8: Distributed / Multi-Agent

**Example:** Multi-agent on GKE with Vertex AI, agent swarms with Pub/Sub coordination, A2A/MCP on GCP

### Governance

- ☐ Require system-level threat modelling covering inter-agent trust, lateral movement, and cascading failure
- ☐ Publish Multi-Agent Security Architecture Standard with GKE namespace-based trust boundaries
- ☐ Mandate zero-trust: unique service account per agent, no shared credentials. Prohibit long-lived service account keys; all agents must use Workload Identity with short-lived tokens. Agents must never be permitted to modify their own identity bindings, IAM policies, or trust relationships under any circumstance
- ☐ Implement kill switches via Cloud Functions modifying Firewall Rules triggered by SCC findings
- ☐ Conduct multi-agent red team exercises in isolated test projects

### Identity

- ☐ Assign each agent a unique GKE Workload Identity service account
- ☐ Implement mTLS between agents using GKE service mesh (Anthos Service Mesh / Istio)
- ☐ Apply IAM deny policies preventing agents from accessing peer agent resources
- ☐ Implement short-lived service account tokens (max 1 hour) for inter-agent auth
- ☐ Audit all inter-agent service account usage in Cloud Audit Logs

### Data

- ☐ Encrypt all inter-agent messages with mTLS via service mesh and optional Cloud KMS envelope encryption
- ☐ Implement signed messages between agents using Cloud KMS asymmetric keys
- ☐ Apply VPC Service Controls preventing data leakage across agent trust domain perimeters
- ☐ Log all inter-agent data exchanges to Cloud Logging in structured format

### Processing (Apps, APIs, Integration)

- ☐ Deploy centralised orchestration using Workflows with human approval callback steps
- ☐ Implement circuit breakers in Workflows and GKE using health checks and PodDisruptionBudgets
- ☐ Isolate each agent in GKE namespace with Kubernetes Network Policies and resource quotas
- ☐ Enforce GKE Pod Security Admission (PSA) at “restricted” level on all agent namespaces: non-root containers, read-only root filesystem, no privilege escalation, no host networking or PID sharing
- ☐ Deploy OPA Gatekeeper or Kyverno admission controllers on GKE to enforce agent-specific workload policies: allowed container registries (Artifact Registry only), required labels and annotations, resource limits, prohibited volume mounts, and mandatory Anthos Service Mesh sidecar injection
- ☐ Apply Pub/Sub message rate limits for inter-agent communication
- ☐ Use Chaos Monkey or Litmus for fault injection testing of multi-agent resilience on GKE

### Network

- ☐ Apply GKE Network Policies preventing lateral movement between agent namespaces
- ☐ Enforce mutual TLS (mTLS) via Anthos Service Mesh for all inter-agent traffic. Where Anthos Service Mesh is not used, implement mTLS via Envoy/Istio-compatible sidecars or equivalent service-mesh architecture. The requirement is cryptographic peer authentication and encrypted east-west traffic – the implementation mechanism may vary. Certificates must be bound to individual agent identities (Workload Identity or pod identity); shared certificates across agents are prohibited. Implement automated certificate rotation with certificate lifetime not exceeding 24 hours for agent identities
- ☐ Segment agent VPCs using VPC peering with firewall rules restricting cross-VPC agent traffic
- ☐ Deploy Cloud Armor and firewall kill switches for rapid agent group isolation
- ☐ Enable VPC Flow Logs with Cloud Logging alerting on anomalous inter-agent traffic patterns

### Supply Chain


© Deep Cyber Ltd 2026

- ☐ Verify agent identity via Workload Identity before mesh admission
- ☐ Enforce Binary Authorization requiring signed and attested images for all mesh agents
- ☐ Scan all agent images with Artifact Registry and Container Analysis
- ☐ Monitor for rogue agents: alert on unregistered service accounts attempting mesh API calls

## Monitoring and Operations

- ☐ Deploy Cloud Monitoring dashboards showing inter-agent Pub/Sub message volumes, errors, and latency
- ☐ Create SCC custom findings for cascade detection: correlated failures across agent namespaces
- ☐ Monitor Cloud Audit Logs for inter-agent trust delegation anomalies
- ☐ Enable SCC Premium and GKE Security Posture for all agent projects
- ☐ Integrate Anthos Service Mesh telemetry into Chronicle for multi-agent SOC visibility
- ☐ Use Litmus or similar for periodic chaos engineering validating multi-agent resilience
- ☐ Alert on unexpected service account creation or Workload Identity binding changes
- ☐ Retain full interaction history in Cloud Storage with bucket lock for forensics


© Deep Cyber Ltd 2026